Unüberwachtes Lernen mit dem Herz-Datensatz

Clustering und PCA auf die Herzdaten

Einlesen der Herz-Daten

Es werden wieder die Herzdaten aus der letzten Aufgabe genutzt. Lesen Sie diese als Data Frame ein.

```
# Ihre Lösung:
herz = read.csv(url("https://oc.informatik.hs-mannheim.de/s/wyzFq34K9HiNjXR/download")) |> mutate(across
herz$ca = as.numeric(as.factor(herz$ca))
herz$goal = factor(herz$goal, levels = c(0, 1, 2, 3, 4), labels = c("gesund", "krank", "krank", "krank"
```

Bedeutet "ähnliche Merkmale" auch "gleiche Diagnose"?

Für jeden Datensatz ist bekannt, zu welcher Klasse er gehört: 0 (gesund) und 1 (erkrankt). Wir wollen untersuchen, wie gut *ähnliche* Datensätze zur gleichen Klasse gehören. Dafür soll mit dem k-means-Clusterverfahren der Datensatz in zwei Cluster eingeteilt werden.

Nur reelle Merkmale

Zunächst sollen **nur die nummerischen Merkmale** benutzt werden und nicht jene, die Faktoren sind.

Clustering Clustern Sie diese Daten. Überlegen Sie, ob Sie die Daten standardisieren wollen.

```
# Ihre Lösung:
nummerische_merkmale = herz |>
  select_if(is.numeric) |> # nur nummerisch
  scale() |>
  as.data.frame()#standardisiert
set.seed(42)
kmeans_result = kmeans(nummerische_merkmale, centers = 2, nstart = 25)
print(kmeans_result)
```

```
## K-means clustering with 2 clusters of sizes 165, 138
##
## Cluster means:
##          age   trestbps       chol    thalach    oldpeak         ca
## 1 -0.5923548 -0.2764543 -0.2025422  0.5046668 -0.4402120 -0.5117952
## 2  0.7082503  0.3305432  0.2421700 -0.6034059  0.5263404  0.6119290
##
## Clustering vector:
##   [1] 2 2 2 1 1 1 2 1 2 2 1 1 2 1 1 1 1 1 1 1 2 1 1 2 2 1 1 2 1 1 2 2 2 1 1 1 1
##  [38] 2 2 2 2 1 2 2 1 2 1 2 2 1 1 1 1 1 2 2 1 1 1 2 1 1 2 1 2 2 2 1 2 2 2 2 2 2
##  [75] 1 2 2 2 1 2 1 1 1 2 1 1 1 1 1 1 2 2 2 1 1 1 2 2 1 1 1 1 1 2 2 1 1 1 2 1 2
## [112] 2 1 2 2 1 1 1 2 2 1 2 1 2 2 1 2 2 1 1 1 1 1 1 1 1 2 2 1 1 1 2 1 2 1 1 2 1
## [149] 1 1 1 1 2 2 2 2 1 2 2 2 1 2 1 1 1 1 1 1 1 1 2 1 2 2 2 2 2 2 1 2 1 2 1 2 1
## [186] 2 1 2 1 2 1 2 1 2 2 2 2 1 1 2 1 2 1 2 1 2 2 1 1 2 1 1 1 2 1 1 1 1 2 1 1 1
## [223] 1 2 2 1 1 2 2 2 1 2 2 2 1 2 2 1 1 1 1 1 1 2 2 2 1 2 1 1 1 2 2 1 1 1 2 2 2
## [260] 1 1 2 1 1 2 1 1 2 1 1 2 2 2 2 2 2 2 1 1 1 2 1 2 1 1 2 2 1 1 1 2 1 1 2 1 1
```

```
## [297] 2 2 1 2 2 1 1
##
## Within cluster sum of squares by cluster:
## [1] 542.0245 842.9371
##  (between_SS / total_SS =  23.6 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

Richtig? Berechnen Sie, wie viel Prozent der Datensätze richtig einem Cluster eingeordnet wurden und geben Sie die Zahl auf zwei Nachkommastellen gerundet aus.

Hinweis: Berücksichtigen Sie, dass die Vergabe der Clusternummern zufällig ist. D.h. sowohl die Cluster (1, 2) wie auch (2, 1) sind möglich.

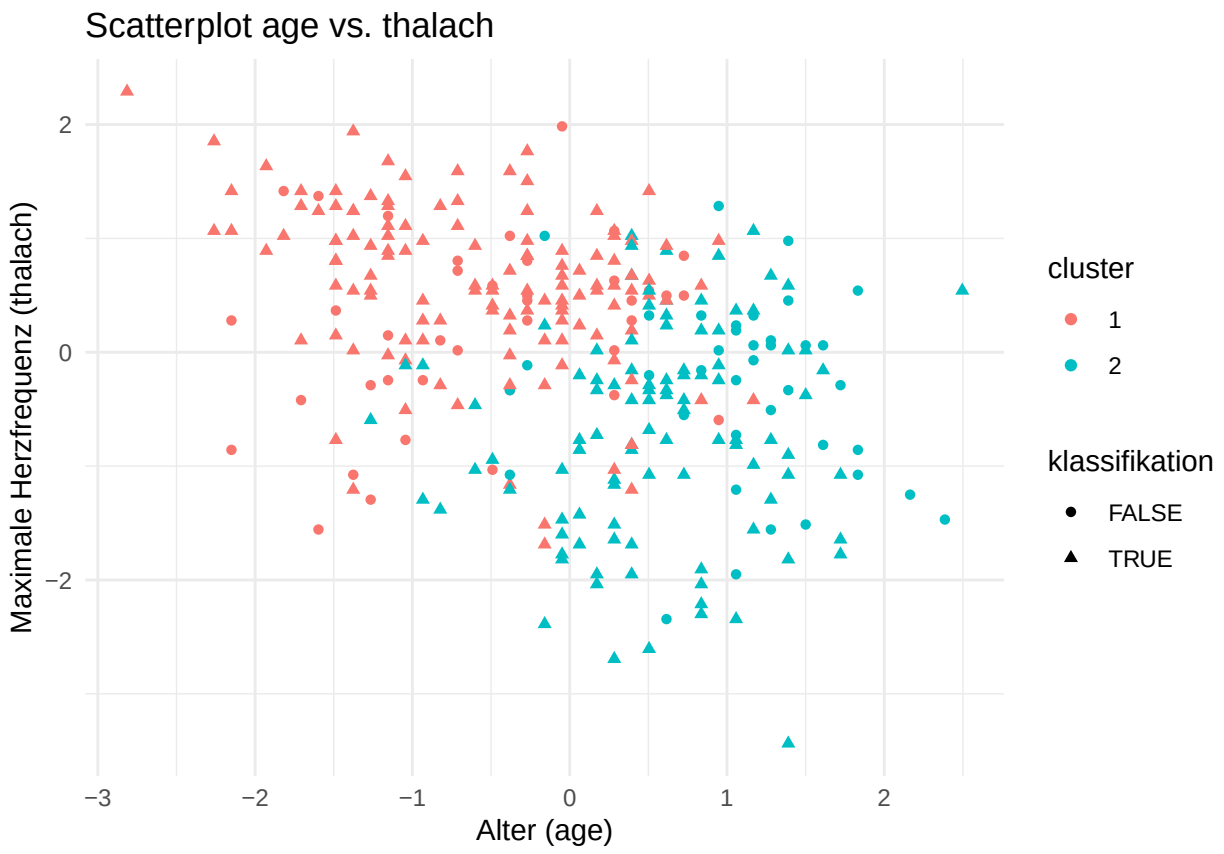
```
# Ihre Lösung:
cluster1 = kmeans_result$cluster |>
  sapply(function(X) X == 1)
cluster2 = kmeans_result$cluster |>
  sapply( function(X) X == 2)
prozentsatz_cluster1 = round(mean(cluster1 == (herz$goal == "gesund")) * 100, digits = 2)
prozentsatz_cluster2 = round(mean(cluster2 == (herz$goal == "gesund")) * 100, digits = 2)

#überprüfen welches Cluster größer ist

if (prozentsatz_cluster1 > prozentsatz_cluster2) {
    korrekte_zuordnungen = cluster1
    prozentsatz_korrekte_zuordnungen = prozentsatz_cluster1
} else {
    korrekte_zuordnungen = cluster2
    prozentsatz_korrekte_zuordnungen = prozentsatz_cluster2
}
print(paste(prozentsatz_korrekte_zuordnungen, "% wurden richtig zugeordnet"))
```

```
## [1] "74.59 % wurden richtig zugeordnet"
```

Scatterplot age vs. thalach Plotten Sie die Merkmale `age` und `thalach` als Scatterplot. Färben Sie die Punkte gemäß ihrer Clusterzuordnung ein. Die Form (`shape`) eines Punkts soll zeigen, ob die Klassifikation (d.h. der Cluster) richtig oder falsch ist.

```
# Ihre Lösung:
# Scatterplot age vs. thalach
nummerische_merkmale$cluster = as.factor(kmeans_result$cluster)
nummerische_merkmale$klassifikation = korrekte_zuordnungen == (herz$goal == "gesund")


scatterplot = ggplot(nummerische_merkmale) +
  geom_point( aes(x = age, y = thalach, color = cluster, shape = klassifikation)) +
  labs(title = "Scatterplot age vs. thalach ",
       x = "Alter (age)",
       y = "Maximale Herzfrequenz (thalach)") +
  theme_minimal()
```

```r
print(scatterplot)
```



Mit Dummy-Variablen

Nun sollen **alle Merkmale** benutzt werden.

Clustering Clustern Sie diese Daten. Überlegen Sie, wie die Faktoren zu Zahlen werden.

```r
# Ihre Lösung:
herz_dummy = herz |>
  mutate_if(function(x) !is.numeric(x), function(y) as.numeric(as.factor(y))) |>
  scale()
herz_dummy = as.data.frame(herz_dummy)


kmeans_result_dummy = kmeans(herz_dummy, centers = 2, nstart = 25)
```

Richtig? Berechnen Sie für diesen Fall, wie viel Prozent der Datensätze richtig einem Cluster eingeordnet wurden und geben Sie die Zahl auf zwei Nachkommastellen gerundet aus. Wie hat sich der Wert verändert? Warum ist dies so?

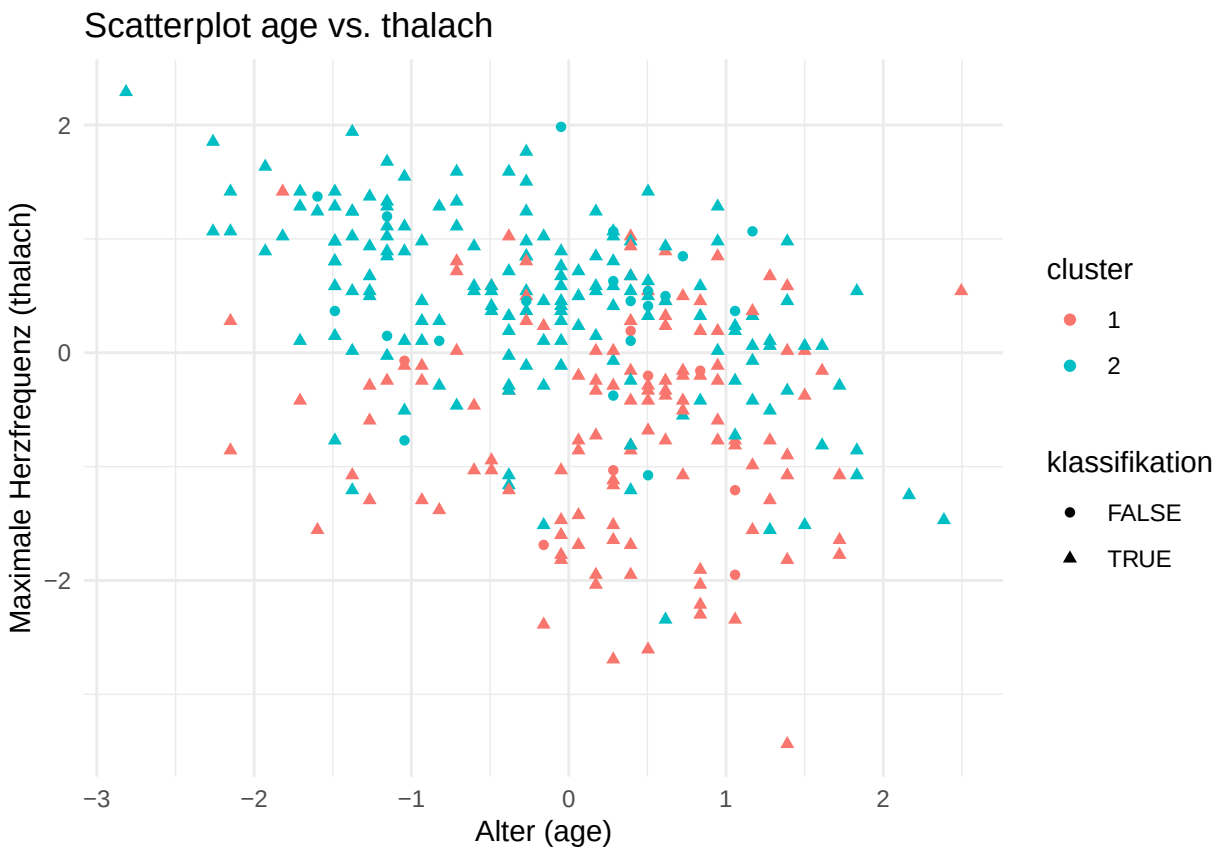
```r
# Ihre Lösung:
cluster3 = kmeans_result_dummy$cluster |>
  sapply(function(X) X == 1)
cluster4 = kmeans_result_dummy$cluster |>
  sapply( function(X) X == 2)
prozentsatz_cluster3 = round(mean(cluster3 == (herz$goal == "gesund")) * 100, digits = 2)
```

```r
prozentsatz_cluster4 = round(mean(cluster4 == (herz$goal == "gesund")) * 100, digits = 2)

#überprüfen welches Cluster größer ist

if (prozentsatz_cluster3 > prozentsatz_cluster4) {
    korrekte_zuordnungen_dummy = cluster3
    prozentsatz_korrekte_zuordnungen_dummy = prozentsatz_cluster3
} else {
    korrekte_zuordnungen_dummy = cluster4
    prozentsatz_korrekte_zuordnungen_dummy = prozentsatz_cluster4
}
print(paste(prozentsatz_korrekte_zuordnungen_dummy, "% wurden richtig zugeordnet"))
```

```
## [1] "90.76 % wurden richtig zugeordnet"
```

Scatterplot age vs. thalach Plotten Sie erneut und schauen Sie, wie die richtigen nun Punkte verteilt sind.

```r
# Ihre Lösung:
herz_dummy$cluster = as.factor(kmeans_result_dummy$cluster)
herz_dummy$klassifikation = korrekte_zuordnungen_dummy == (herz$goal == "gesund")


scatterplot = ggplot(herz_dummy) +
  geom_point( aes(x = age, y = thalach, color = cluster, shape = klassifikation)) +
  labs(title = "Scatterplot age vs. thalach ",
       x = "Alter (age)",
       y = "Maximale Herzfrequenz (thalach)") +
  theme_minimal()

print(scatterplot)
```



PCA

Wenden Sie eine PCA auf diesen Datensatz an. Es sollen alle Merkmale berücksichtigt werden.

Wichtige Merkmale

Welche Merkmale der ersten Hauptkomponente tragen am meisten zur Varianz bei? Geben Sie die TOP-10-Merkmale an.

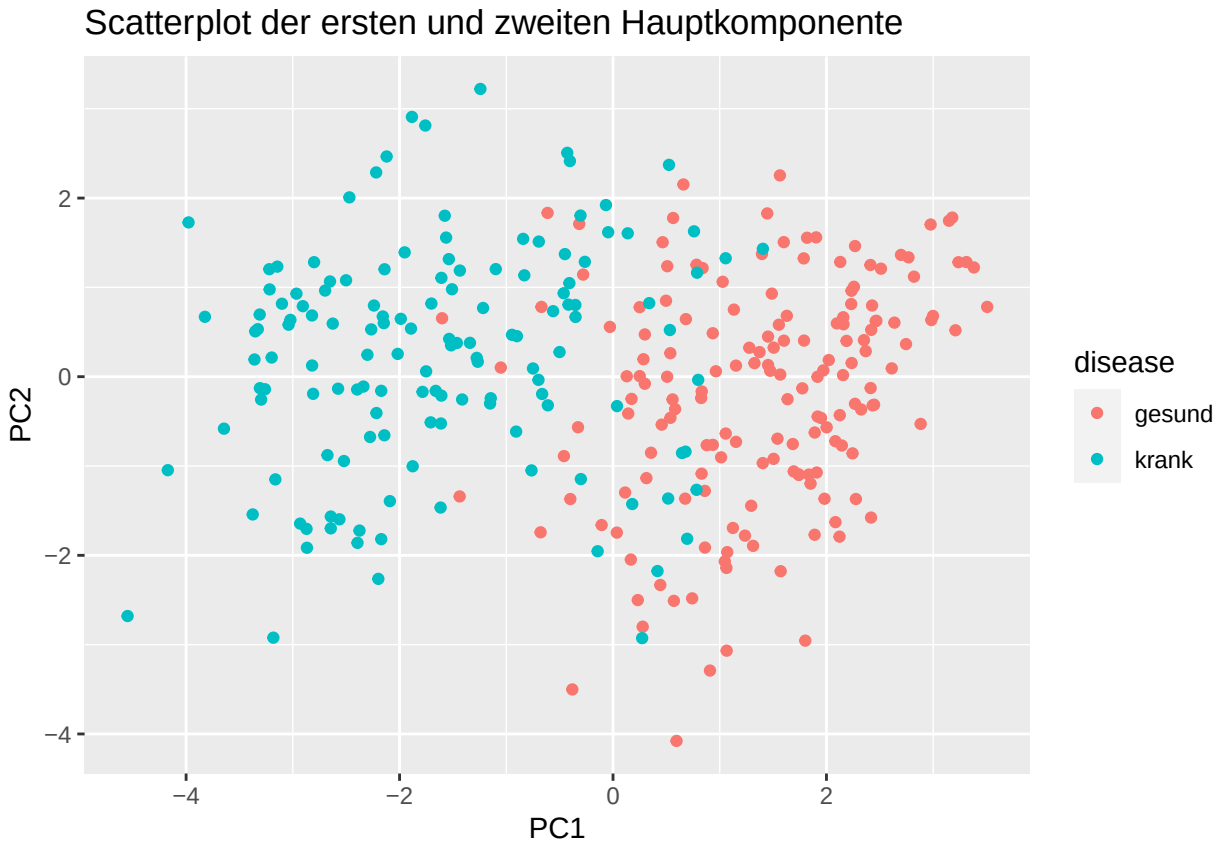
```
# Ihre Lösung:
herz_pca = herz_dummy |>
  mutate(across(where(is.numeric), scale))

pca = prcomp(select(herz_pca, where(is.numeric)), center = TRUE, scale. = TRUE)
top_10 = pca$rotation[ , 1] |>
  abs()

top_10 = head(sort(top_10, decreasing = TRUE), 10)

print(top_10)
```

```
##      goal   oldpeak   thalach      thal     exang     slope        ca        cp
## 0.4179508 0.3554540 0.3468915 0.3235372 0.3114819 0.3083485 0.2929885 0.2734216
##       age  trestbps
## 0.2413238 0.1392466
```

Erste und zweite Hauptkomponente

Plotten Sie die erste und zweite Hauptkomponente als Scatterplot. Färben Sie die Punkte gemäß ihrer Klasse (Disease) ein.

```
# Ihre Lösung:
scatter_pca = as.data.frame(pca$x[, 1:2])
scatter_pca$disease = factor(herz$goal)
ggplot(scatter_pca, aes(PC1, PC2, color = disease)) +
  geom_point() +
  labs(title = "Scatterplot der ersten und zweiten Hauptkomponente",
       x = "PC1",
       y = "PC2")
```

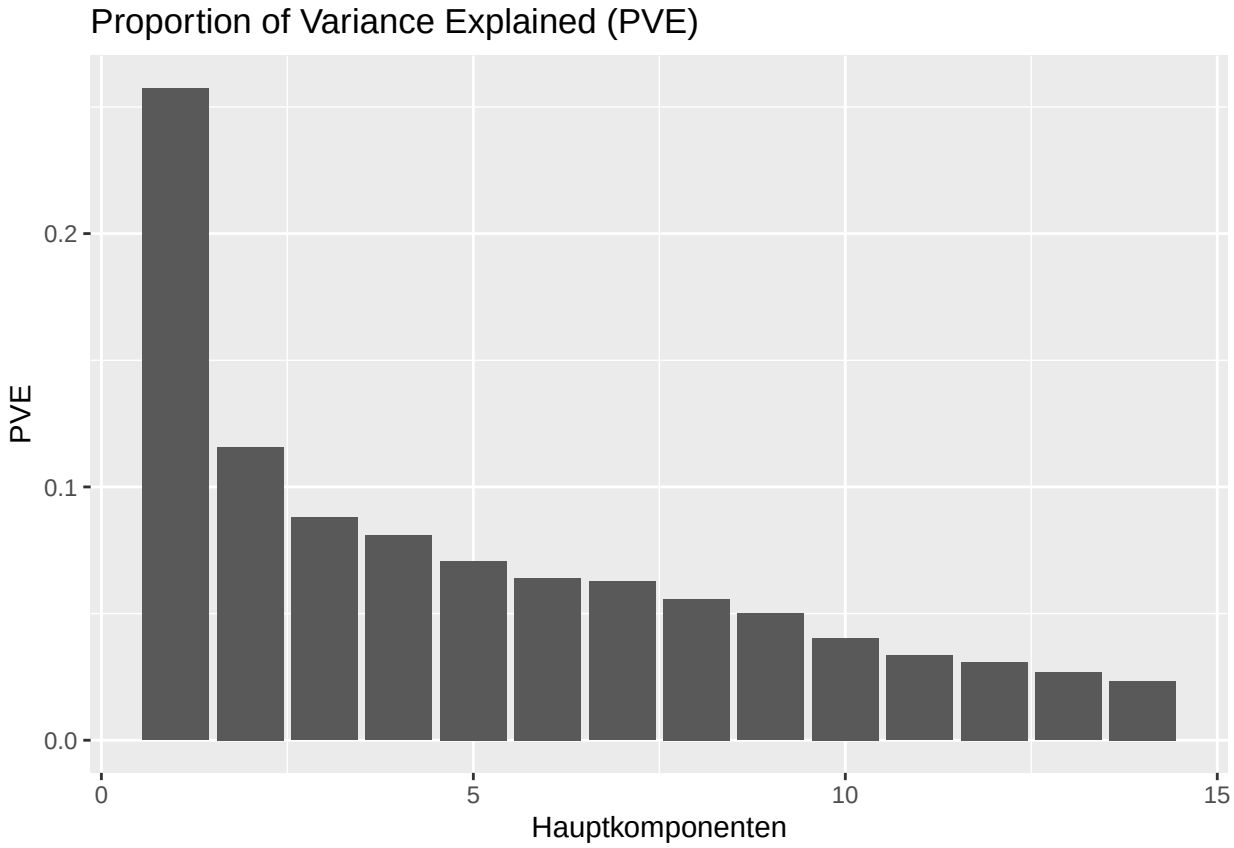


PVE

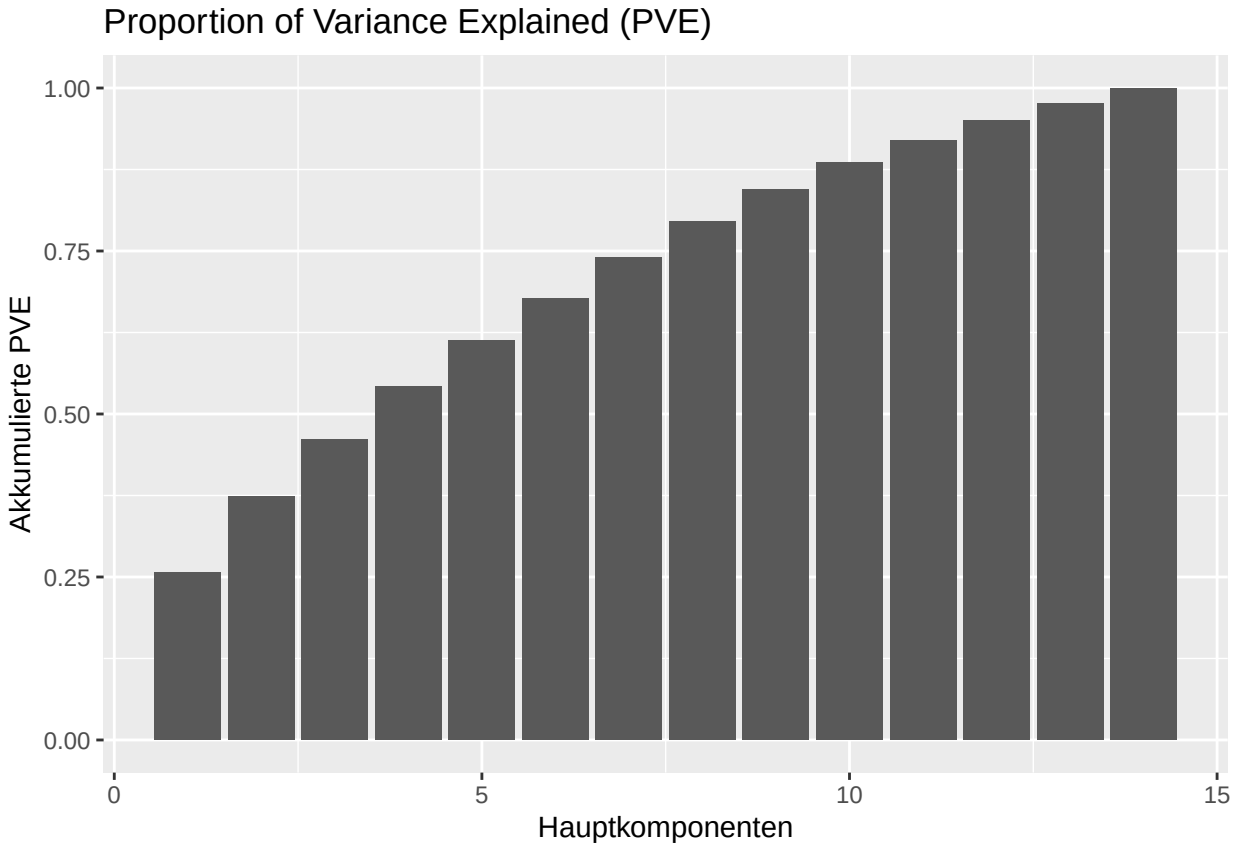
Plot Plotten Sie die Proportion of Variance explained (PVE) für jede Hauptkomponente sowie die akkumulierte PVE.

```
# Ihre Lösung
pve = (pca$sdev^2) / sum(pca$sdev^2)
pve_daten = data.frame(
  PC = seq_along(pve),
  PVE = pve,
  akk_PVE = cumsum(pve)
)

ggplot(pve_daten) +
  geom_bar(aes(x = PC, y = PVE), stat = "identity") +
```

```
  labs(title = "Proportion of Variance Explained (PVE)",
       x = "Hauptkomponenten",
       y = "PVE")
```




```
ggplot(pve_daten, aes(x = PC)) +
  geom_bar(aes( x = PC, y = akk_PVE ), stat = "identity") +
  labs(title = "Proportion of Variance Explained (PVE)",
       x = "Hauptkomponenten",
       y = "Akkumulierte PVE")
```



Wichtige Hauptkomponenten Wie viele Hauptkomponenten erklären mehr als 50% der Varianz?

Möglicherweise tragen bei Ihrem Ergebnis die letzten Hauptkomponenten keine Varianz mehr bei. Überlegen Sie, woran das liegen könnte.